Institute of Architecture of Application Systems

Comparing Quantum Service Offerings

Obst, Julian and Barzen, Johanna and Beisel, Martin and
Leymann, Frank and Salm, Marie and Truger, Felix

Institute of Architecture of Application Systems
University of Stuttgart
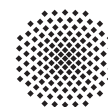Universitätsstraße 38, 70569 Stuttgart, Germany
http://www.iaas.uni-stuttgart.de

BIBTEX:

```
@inproceedings{Obst2023_ComparingQuantumOfferings,
    Title          = {{Comparing Quantum Service Offerings}},
    Author         = {Obst, Julian and Barzen, Johanna and Beisel, Martin and Leymann, Frank and
                      Salm, Marie and Truger, Felix},
    Year           = 2023,
    Month          = sep,
    Booktitle      = {2023 IEEE International Conference on Quantum Computing and Engineering (QCE)},
    Publisher      = {IEEE},
    Pages          = {181--184},
    Volume         = {02},
    Doi            = {10.1109/QCE57702.2023.10206},
    url            = {https://ieeexplore.ieee.org/document/10313655},
    projects       = {PlanQK,EniQmA,SeQuenC},
}
```

Universität Stuttgart
Germany

II. BACKGROUND AND RELATED WORK

There exist different physical realizations of qubits such as ion traps, quantum dots, or superconducting qubits. Our experiments use quantum devices with superconducting qubits or trapped ions, being common quantum technologies [8].

We evaluate QAOA [7] and consider an instance of the *Max-Cut* problem, where the goal is to divide a weighted graph's vertices into two partitions to maximize the total weight of edges with endpoints in different partitions. MaxCut was considered in the seminal QAOA paper [7] and is widely used as a benchmark since [9], [10]. QAOA is a *Variational Quantum Algorithm (VQA)*, i.e., it employs a classical optimizer to train a parameterized quantum circuit [11]. The circuit comprises two sub-circuits parameterized by angles γ, β, executed subsequently. The optimizer iteratively improves the values of γ, β, based on previous results. QAOA's depth can be adjusted via a hyperparameter p, specifying the number of repetitions of the sub-circuit pair. Limitations of NISQ devices require low p, e.g., $p = 1$ or $p = 2$ (depth-1, depth-2 QAOA), yet, QAOA can provide meaningful results, making it suitable for the NISQ era. The number of adjustable parameters increases with p, e.g., depth-2 QAOA, optimizes $\gamma_1, \gamma_2, \beta_1, \beta_2$. With increasing p, the algorithm converges towards an optimum [7].

Pelofske et al. [10] and Baker and Radha [9] benchmark quantum devices using QAOA. They compare the balance of the optimal solution and perform mean-variance portfolio optimization, respectively. These benchmarks focus on the accuracy of results, our work compares devices through experiments and focuses on the development, access, and execution processes using QCaaS offerings, outlining encountered issues and observations to support developers with knowledge about typical pitfalls and obstacles. Unlike Vietz et al. [12], who focus on engineering quantum algorithms for the cloud, i.e., composition and integration, our work concentrates on the obstacles during implementation for different QCaaS offerings and comparing experimental results from different devices.

Fig. 1: Procedure of developing, executing, and evaluating QAOA circuits.

Abstract—With the emergence of quantum computing, a growing number of quantum devices is accessible via cloud offerings. However, due to the rapid development of the field, these quantum-specific service offerings vary significantly in capabilities and requirements they impose on software developers. This is particularly challenging for practitioners from outside the quantum computing domain who are interested in using these offerings as parts of their applications. In this paper, we compare several devices based on different hardware technologies and provided through different offerings, by conducting the same experiment on each of them. By documenting the lessons learned from our experiments, we aim to support developers in the usage of quantum-specific offerings and illustrate the differences between predominant quantum hardware technologies.

Index Terms—Quantum Computing, Quantum Cloud Offerings, QAOA

I. INTRODUCTION

Quantum computing promises a computational advantage over its classical counterpart in various fields, such as optimization and chemistry [1], [2]. However, the capabilities of current *Noisy Intermediate-Scale Quantum (NISQ)* devices are still limited by low numbers of qubits and high error rates [3], [4]. To improve their quality, various approaches to implement the qubits of a quantum device, e.g., trapped ions [5] or superconducting electrical circuits [4], are currently being explored. To enable easy access to quantum devices, they are typically provided via the cloud by so-called *Quantum Computing as a Service (QCaaS)* offerings [6].

However, the heterogeneity of these QCaaS offerings, as well as differences of the available quantum devices, impose numerous problems to quantum software developers. These include designing the quantum algorithm to meet hardware capabilities, the implementation and execution of the resulting quantum circuits, and the evaluation of execution results. Overcoming these issues is particularly difficult for practitioners from outside the quantum domain, as existing offerings and tools are tailored for domain experts. To prepare unfamiliar developers for the problems imposed by current quantum devices and QCaaS offerings in practice, we execute the well-known *Quantum Approximate Optimization Algorithm (QAOA)* [7] on four devices provided through different QCaaS offerings and (i) describe the issues we encountered during this process. Moreover, we (ii) evaluate and compare the execution results by employing the mean absolute difference as a metric to assess their noisiness.

III. Experiments and Encountered Issues

In our experiments, we follow the process depicted in Figure 1 to examine QAOA for MaxCut on a selection of quantum devices and evaluate the optimization landscape of QAOA's variational parameters. This approach enables a more objective comparison of devices, avoiding uncertainties from parameter optimization. The optimization landscape is a visualization of the objective function. Here, the objective function is the expected value of a cut that is the result of a measurement. While these landscapes seem similar to plots of complex analysis [13], the landscapes here are not complex functions. To create the landscape, we sample equidistant points in the parameter space of depth-1 and depth-2 QAOA. For each sampled point (1000 shots), we evaluate the expectation value in order to interpolate the optimization landscape. Due to resource limitations, we restrict the sampled parameter space to $(\beta, \gamma) \in ([0, \frac{\pi}{2}], [0, \pi])$, where each sample is $\frac{\pi}{20}$ apart. In the depth-2 case, we fix β_1, γ_1 to their optimal values found in the depth-1 case to visualize the 2-dimensional (β_2, γ_2)-optimization landscape. We thereby follow the idea of a layer-wise parameter initialization that can be seen as a warm-start of depth-2 QAOA utilizing the result from depth-1 QAOA [14]. The experiments were run on four devices, all manufactured by different vendors, and a simulator for comparison with error-free results. Due to confidentiality reasons and the limited scope of our experiments, the names of the offerings are omitted to avoid premature conclusions.

A. Selecting the Problem Instance

In the first step, shown in Figure 1, an instance of the MaxCut problem, i.e., a graph, needs to be selected. As the complexity of typical real-world problems exceeds the capabilities of current quantum devices, we chose a small graph (see Figure 1), which is suitable for NISQ devices.

Connectivity Requirements. The structure of the problem instance already imposes issues, as it requires certain qubit connections for the execution on a device. For the MaxCut problem tackled with QAOA, the targeted graph directly shows the required connectivity, since each node of the graph will be mapped to one qubit and each edge of the graph introduces two-qubit gates between the adjacent qubits. The resulting circuit is depicted in Figure 2. Our problem instance requires a moderate connectivity with 5 out of 10 possible connections between 5 nodes, and is thus feasible for devices ranging between low and full connectivity.

B. Implement & Compile Circuits

In the second step of Figure 1, the QAOA circuits are implemented and compiled for the quantum device they shall be executed on. In the following, we discuss the issues encountered during the implementation and compilation.

Heterogeneous Ecosystems and Formats. The lack of an established vendor-independent language for implementing quantum algorithms complicates their development for developers who do not want to restrict themselves to a specific ecosystem, risking a vendor lock-in. Although there are tools to translate between different formats [15], [16], they often only support specific releases of the frequently updated quantum SDKs and do not always provide a semantically equivalent result [17]. Thus, despite the availability of various translation tools, we had to manually adapt our implementation for individual QCaaS offerings.

Compilation Quality. As most quantum devices have a restricted connectivity between their qubits and support only a subset of quantum gates natively, quantum circuits must be compiled for the target device [3]. This process is NP-hard [18]; thus, the compilation results can differ significantly, e.g., in circuit depth or the number of two-qubit gates [19]. Most offerings give insight into the compiled circuits, such that the compilation process can be repeated multiple times, enabling their comparison and selection for execution. However, one of the offerings used for our experiments encrypts the compilation results, making an analysis and comparison infeasible.

C. Execute Circuits

After implementing the quantum circuit addressing the chosen MaxCut problem and compiling it for the different devices, it can be executed. However, the execution step imposes various offering-specific issues.

Execution Endpoints. The execution endpoints of the different QCaaS offerings differ in many aspects: First, some offerings do not support batching multiple circuits, e.g., for sampling a parameter space, as described above, into a single execution job. Hence, developers must process all circuit execution requests and responses individually, making it an error-prone process. Another issue is that one offering uses different formats for simulated and real quantum devices. Hence, developers cannot be certain that their implementations tested with a simulator will also work on a costly quantum device. Another inconvenience that we observed when executing circuits using pay-per-use QCaaS offerings, is the difficulty of estimating the execution costs, particularly for VQAs, as the number of iterations varies.

Execution Time. Waiting times significantly exceeded the execution times in our experiments. Thus, different options for accessing the devices must be considered to reduce waiting times. For example, queuing can lead to waiting times of multiple hours. However, some offerings also provide exclusive time slot reservations or high-priority access. In the context of reservations, we noticed that it is important to efficiently use the usually short time slots of a few minutes, e.g., by pre-compiling circuits and avoiding extensive classical processing between circuit executions. One device that is still in experimental stage and human-operated could not be automatically accessed via an execution endpoint.

D. Evaluate Results

After executing the circuit and obtaining the measurement results, the final step shown in Figure 1 entails their evaluation.

Result Format. The offerings' responses differ significantly. In particular, the measurement results are returned in different
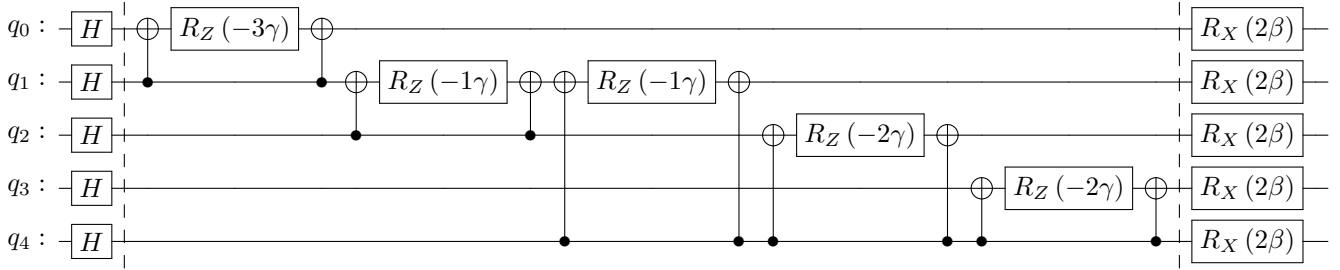
Fig. 2: QAOA circuit of our instance. The final measurement operations are omitted. Note that each edge of the graph is represented as an R_Z-gate between two CNOT-gates.

IV. EXPERIMENT RESULTS

The results of our experiments are illustrated in Figure 3. Their quality varies significantly across the four devices, and particularly across devices based on different hardware technologies. The contour plots depict the objective function for QAOA in the so-called optimization landscape. As explained above, it is a visualization of the function that the optimizer tries to minimize. Columns correspond to simulation and the different devices, and rows to two replications of depth-1 and one run for depth-2 QAOA. Low (blue) values are desirable, as they resemble lower energy, corresponding to better solutions of the MaxCut problem. In contrast, high (red) values indicate unfavorable parameter values.

The fact that result quality can change over time is demonstrated by the two replications for depth-1 QAOA per offering. Thus, repeating experiments at a later point with updated device calibrations can affirm the validity of results. Evidently, various error sources, deteriorate the results significantly compared to the simulation in the leftmost column of Figure 3. These results also illustrate the impact of the circuit depth, as the landscapes for depth-2 QAOA in the bottom row appear less similar to the simulation compared to those for depth-1 in the upper rows. These insights are underlined by the *Mean Absolute Differences (MADs)* shown in Figure 4, which are calculated by averaging the absolute differences between the samples of two landscapes:

$$\text{MAD}(E_1, E_2) = \frac{1}{|\Gamma| \cdot |B|} \sum_{(\gamma, \beta) \in \Gamma \times B} |E_1(\gamma, \beta) - E_2(\gamma, \beta)|$$

$\Gamma = \{0, \frac{\pi}{20}, \dots, \frac{19\pi}{20}, \pi\}$ and $B = \{0, \frac{\pi}{20}, \dots, \frac{9\pi}{20}, \frac{\pi}{2}\}$ are the samples for γ and β respectively, and E_1, E_2 are the expectation values of the objective function, also called energy, for two different landscapes. The MAD is an intuitive measure for the difference of two ordered sets. It can be used to assess the performance of a model [20]. The MAD_{SIM} serves as a similarity measure between landscapes obtained from the quantum devices and the simulation, whereas the MAD_{MMS} is a similarity measure between landscapes obtained from the devices and the *Maximally Mixed State (MMS)*, i.e., the mixture where every basis state is equally probable. The MMS resembles a random sampling of bit strings, therefore, the MAD_{MMS} is an indicator of noise [21].

Both Figures 3 and 4 show a clear difference between the two types of devices. While all devices found a minimum in relative proximity to the true minimum in the depth-1 case, the landscapes produced by the trapped ion devices resemble the simulation much closer. This becomes even more evident in the depth-2 case, where the result of device 2 is dominated by noise. As the MAD_{SIM} in Figure 4 shows, the increased QAOA depth also led to a reduced similarity to the simulation. For the ion trap devices, the MAD_{MMS} in Figure 4 increased slightly with the QAOA depth. A greater MAD_{MMS} indicates a less noisy result. Evidently, ion trap devices performed better in our experiments. A possible explanation is the graph density of the problem instance. Pelofske et al. [10] also found that trapped ion devices performed better for instances requiring a high connectivity. However, our experiments only cover one use case, hence, other circuit types may lead to different results.

V. SUMMARY AND FUTURE WORK

In this work, we documented the lessons learned using different QCaaS offerings. To this end, we implemented and executed the same experiment, addressing the MaxCut problem with QAOA, for four quantum devices based on ion-trap and superconducting qubits. The heterogeneity of the QCaaS offerings complicated the development and execution. In particular, differences in formats, access options, and limitations, e.g., regarding metadata, had to be considered. Additionally, noise and errors of current quantum devices deteriorated the results. However, the impact differs depending on the underlying hardware technology. In our experiments,
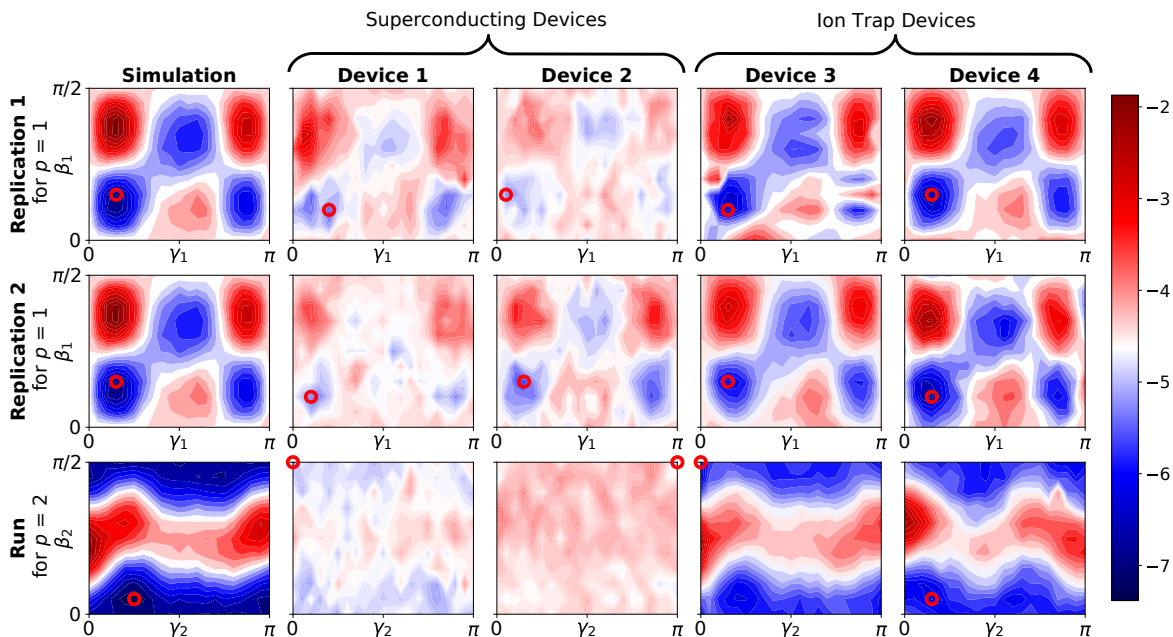
Fig. 3: Plotted energy expectation in the (γ_1, β_1) and (γ_2, β_2) parameter spaces. Measured minima are marked by red circles. The first two rows show results for depth-1 QAOA, replicated on different days. The bottom row shows results for depth-2 QAOA, with parameters β_1, γ_1 fixed to optimal values from replication 2.
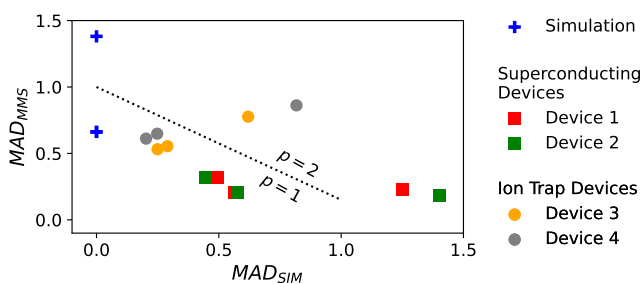

Fig. 4: MADs of the energy expectation values plotted in Figure 3. A low MAD_{SIM} and a high MAD_{MMS} are favorable.

the results obtained from superconducting devices were significantly more erroneous compared to those of ion trap devices.

In future work, we plan to address the heterogeneity of QCaaS offerings by means of a unified access layer, facilitating device-agnostic development and execution. Furthermore, we will expand our experiments to more use cases, quantum algorithms, quantum devices, and hardware technologies.

REFERENCES

[1] M. P. Harrigan *et al.*, "Quantum approximate optimization of non-planar graph problems on a planar superconducting processor," *Nature Physics*, vol. 17, no. 3, pp. 332–336, 2021.

[2] Y. Cao, J. Romero, and A. Aspuru-Guzik, "Potential of quantum computing for drug discovery," *IBM Journal of Research and Development*, vol. 62, no. 6, pp. 6:1–6:20, 2018.

[3] F. Leymann and J. Barzen, "The bitter truth about gate-based quantum algorithms in the NISQ era," *Quantum Science and Technology*, pp. 1–28, 2020.

[4] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, 2018.

[5] C. D. Bruzewicz, J. Chiaverini, R. McConnell, and J. M. Sage, "Trapped-ion quantum computing: Progress and challenges," *Applied Physics Reviews*, vol. 6, no. 2, p. 021314, 2019.

[6] F. Leymann *et al.*, "Quantum in the Cloud: Application Potentials and Research Opportunities," in *CLOSER'20*. SciTePress, 2020, pp. 9–24.

[7] E. Farhi, J. Goldstone, and S. Gutmann, "A Quantum Approximate Optimization Algorithm," *arXiv:2202.06782*, 2014.

[8] J. D. Whitfield, J. Yang, W. Wang, J. T. Heath, and B. Harrison, "Quantum computing 2022," *arXiv:2201.09877*, 2022.

[9] J. S. Baker and S. K. Radha, "Wasserstein Solution Quality and the Quantum Approximate Optimization Algorithm: A Portfolio Optimization Case Study," *arXiv:2202.06782*, 2022.

[10] E. Pelofske, J. Golden, A. Bärtschi, D. O'Malley, and S. Eidenbenz, "Sampling on NISQ Devices: "Who's the Fairest One of All?"," in *2021 IEEE QCE*, 2021, pp. 207–217.

[11] M. Cerezo *et al.*, "Variational quantum algorithms," *Nature Reviews Physics*, vol. 3, no. 9, pp. 625–644, 2021.

[12] D. Vietz *et al.*, "An Exploratory Study on the Challenges of Engineering Quantum Applications in the Cloud," in *Q-SET 2021*. CEUR Workshop Proceedings, 2021, pp. 1–12.

[13] E. Wegert and G. Semmler, "Phase plots of complex functions: a journey in illustration," *Notices of the American Mathematical Society*, vol. 58, pp. 768–780, 06 2011.

[14] F. Truger *et al.*, "Warm-Starting and Quantum Computing: A Systematic Mapping Study," *arXiv:2303.06133*, 2023.

[15] Quantastica, "Quantum programming language converter," 2023, https://github.com/quantastica/qconvert.

[16] S. Sivarajah *et al.*, "t|ket⟩: a retargetable compiler for nisq devices," *Quantum Science and Technology*, vol. 6, no. 1, p. 014003, 2020.

[17] Y. Kharkov, A. Ivanova, E. Mikhantiev, and A. Kotelnikov, "Arline Benchmarks: Automated Benchmarking Platform for Quantum Compilers," *arXiv:2202.14025*, 2022.

[18] M. Y. Siraichi, V. F. d. Santos, C. Collange, and F. M. Q. Pereira, "Qubit Allocation," ser. CGO 2018. ACM, 2018, pp. 113–125.

[19] M. Salm *et al.*, "Automating the Comparison of Quantum Compilers for Quantum Circuits," in *SummerSOC'21*. Springer, 2021, pp. 64–80.

[20] C. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *Climate Research*, vol. 30, pp. 79–82, 2005.

[21] S. Wang *et al.*, "Noise-induced barren plateaus in variational quantum algorithms," *Nature Communications*, vol. 12, no. 1, 2021.